Installation instructions

These instructions guide the installation of the support systems for IheOs XDS server, Registry and Repository. This installation uses pre-complied code so is static by design. The installation is done in two phases: support systems, the focus of this download, which instructs in the installation of the PostgreSQL database and Tomcat web servers. The second part, the actual server code installation, is contained in a separate download with its own installation instructions.

The installation of the support systems, what you are reading about now, is done once. Once complete, the actual server code can be installed and updated without repeating this process.

These instructions are based on a Unix-like platform. They have been tested on MAC OS 10.4 and Ubuntu Linux. All commands shown are issued into a normal shell (command) window. Shell commands and the responses are shown in `Courier font`. This package will run on Windows but we do not have experience on that platform to offer any guidance for its installation and use.

Prerequisites

PostgreSQL database - Version 8.1 is what we currently use but almost any version will work. This package assumes that PostgreSQL is listening on port 5432 which is the default.

Java 5 JDK (JRE is not sufficient)

It is assumed that you can install PostgreSQL and Java 5 and these instructions assume that is done. These instructions assume that the PostgreSQL tools (bin directory) is in your PATH and the JAVA_HOME environment variable is set.

When testing the installation on Ubuntu, we used the pre-compiled version of PostgreSQL, installing with the command:

```
==> sudo apt-get install postgresql-8.1
```

Installation Overview

The installation has these main pieces:

1. Create supporting databases
2. Install tomcat servers. This is done from the package xds_support_bin_1.tgz
3. Install the XDS server code into tomcat.
4. Starting the services
5. Download/install the xdstest2 test tool and supporting test data
6. Test the installation

User Accounts

We use a single user account for both the PostgreSQL database and Tomcat web engine. PostgreSQL will not run in a system (root) account and Tomcat should not. The easiest way to install is to pick a single account and install all the software in that account. I use my normal login account for development work. In the instructions below, all work in done in this account unless otherwise specified.

If you install on Ubuntu as described above, PostgreSQL will be installed in a new account postgres. If you install this way, create a role for your user account to simplify the use of PostgreSQL command line usage. Without creating a role for yourself, you will have to issue PostgreSQL commands like this:

```
==> sudo -u postgres createdb registry
```

By creating a role for your user account (bill in my case):

```
==> sudo -u postgres createrole -s bill
```

Also create the role bill since I have not yet cleaned its presence out of the database files:

```
==> sudo -u postgres createrole -s bill
```

I can then issue PostgreSQL commands like this:

```
==> createdb registry
```

The instructions below were tested on Ubuntu Linux.

Downloads

A single tar-ball style download contains the binary copy of the XDS Server software and data. Download the tar-balls:

```
==> wget http://ihexds.nist.gov:9080/XdsDocs/opensource/xds_support_bin_1.tgz
```

Before downloading this, I suggest examining the directory for more recent versions of the support package.

Expand the contents:

```
==> tar xfz xds_support_bin_1.tgz
```

which creates the directories xds_support_bin_1

Create Roles and Databases in PostgreSQL

These Roles/Users are used by the various databases created below. Enter the following PostgreSQL commands.

```
==> createuser -s -P ebxmlrr
when prompted for password, hit <RETURN> without entering a
password (twice)
==> createuser -s -P xdsoper
when prompted, give password xdsoper (twice)
==> createuser -s postgres
this may fail because it already exists.  That's ok.
==> createuser -s -P logs
when prompted, give password xdslogs
```

The registry database holds the ebxmlrr registry contents. The adt database holds the ADT patient registration details. The logs database holds the test log. Now create the databases.

```
==> createdb -O xdsoper registry
==> createdb adt
==> createdb -O logs log2
```

Initialize the databases

The databases are initialized from data files in this package.

```
==> pg_restore -d registry -F t init_data/xds/xds_init.tar
==> pg_restore -d adt init_data/adt/adt.dump
```

The log2 database will be initialized by the server the first time there is something to log.

Install Tomcat servers

This system requires two copies of Tomcat to run. The underlying registry is ebxmlrr-2.1-final-1 which is dependent on some very old XML libraries (jar files). These jar files are special in the Java world. There are considered more of a language extension than just a library file. The must be installed in Tomcat as 'endorsed' libraries and all applications installed in tomcat must use the same copies. We use two copies of tomcat each running XML libraries from a different era.

Both copies of tomcat have predefined locations in the filesystem and some code relies on them being in the correct place. If you wish to install elsewhere then create symbolic links so they effectively exist in the predefined places.

Tomcat2
This is the older environment and it holds the ebxmlrr servlet which is the old ebxmlrr -2.1-final-1 ebXML Registry engine. It runs on port 8080.

The only application/servlet running in this copy of Tomcat is ebxmlrr, the back end registry engine. It comes pre-configured in this copy of tomcat.

To install...

This package will be installed in a user account. We refer to the user account name as **user** and the associated group name as **group**.

==> sudo mkdir /usr/local/tomcat2
==> sudo chown **user** /usr/local/tomcat2
==> cd xds_support_bin_1/tomcat2
==> cp -R * /usr/local/tomcat2
==> cd /usr/local/tomcat2
==> chown -R **user** *
==> chgrp -R **group** *

Tomcat1
This is the newer servlet environment and all the rest of our software runs here. It uses port 9080 which is pre-configured.

In this tomcat runs the following servlets:
axis2 - this is the Axis2 web services implementation. More on what runs inside this below.
LogReader - Test Log reader servlet

xdsref - XDS reference materials (Schema etc, Code tables etc)
xdstools - various tools for the XDS environment.

To install...

==> sudo mkdir /usr/local/tomcat1
==> sudo chown **user** /usr/local/tomcat1
==> cd xds_support_bin_1/tomcat1
==> cp -R * /usr/local/tomcat1
==> cd /usr/local/tomcat1
==> chown -R **user** *
==> chgrp -R **group** *

System startup

The system consists of three major components, PostgreSQL, tomcat1 and tomcat2. PostgreSQL must be started before tomcat2. If PostgreSQL must be restarted for any reason, both tomcat1 and tomcat2 must be restarted. The restart order for the tomcats is not important.

How to start PostgreSQL depends on your installation type, from sources or from pre-bundled package. Starting and stopping the tomcats is done by the following commands. Do not start tomcat yet. The instructions below will tell when it is appropriate.

tomcat2 start
    cd /usr/local/tomcat2/bin
    ./startup.sh

tomcat2 stop
    cd /usr/local/tomcat2/bin
    ./shutdown.sh

Following the log output can be done by
    cd /usr/local/tomcat2/logs
    tail -f catalina.out

tomcat1 start
    cd /usr/local/tomcat1/bin
    ./startup.sh

tomcat1 stop
    cd /usr/local/tomcat1/bin

./shutdown.sh

Following the log output can be done by
cd /usr/local/tomcat2/logs
tail -f catalina.out

Installing the XDS Server code

The XDS Server code is downloaded and installed separately. The initial version can be obtained by:

```
==> wget
http://ihexds.nist.gov:9080/XdsDocs/opensource/xds_server_bin/xds_server_bin_
1_0.tgz
```

Always grab the latest version available.

Un-bundle the tar-ball by:

```
==> tar xfz xds_server_bin_1_0.tgz
```

which creates the directory xds_server_bin_1_0. To install, be sure you are using the same account that tomcat1 lives in and...

```
==> cd xds_server_bin_1_0

==> ./install.sh
```

This will install:

- xds.aar - the XDS Registry Adaptor into /usr/local/tomcat1/webapps/axis2/WEB-INF/services
- xdsref.war - the XDS reference materials (Schema, Code tables etc) into /usr/local/tomcat1/webapps
- xdstools.war - toolkit into /usr/local/tomcat1/webapps
- LogReader.war - the Test Log reader into /usr/local/tomcat1/webapps

Later when you need them, the toolkit can be opened in your web browser from

http://localhost:9080/xdstools/

and the Log Reader can be opened in your web browser from

http://localhost:9080/LogReader

Starting the services

Now start the services in the following order:

1. PostgreSQL
2. tomcat2
3. tomcat1

Examine the logs to see they started correctly.

Testing the installation

The xdstest2 toolkit and test data are used to test the installation. These must be downloaded separately from

http://ihexds.nist.gov:9080/XdsDocs/xdstest2/

and

http://ihexds.nist.gov:9080/XdsDocs/testkit/

Always grab the latest release. BTW, ihexds.nist.gov is an alias for the old hcxw2k1.nist.gov name and a lot easier to type. The raw IP address is 129.6.24.109.

Once xdstest2 and testkit are installed and working, they will be used to test the installation. In the following instructions we presume that you are testing from the same machine that the xds service is installed on. If not, the localhost name will have to altered to fit your environment.

Go to the section 11990/submit of the testkit. Edit the <EndPoint/> to point to localhost at port 9080 and run the test. If all is going well, it should be successful. This has excercised the Provide and Register.b transaction to the Repository which in turn generated a Register.b transaction to the Registry.

Then go to the section 11990/eval of testkit. Again edit the <EndPoint/> to point to localhost at port 9080 and run this test. it exercises the GetSubmissionSetandContents Stored Query to retrieve the just submitted metadata from the Registry. Hopefully this works too.

One major thing that could have gone wrong is that the PatientId in the submission could be rejected by the Registry. As delivered, the registry is configured to not validate

Patient IDs. In future releases or upgrades this could change. At this point you may need to register a Patient ID or disable Patient ID validation.

Allocating a Patient ID

This Registry implementation does not accept a real Patient identity Feed transaction. Instead, it uses a web page to allocate Patient IDs. Point your web browser to
http://localhost:9080/xdstools/
and select the link labeled 'Allocate a Patient ID for the Public Registry'
This will:

1) Allocate a new Patient ID
2) Stuff it into the adt database in PostgresSQL
3) Display the Patient ID in the resulting web page

Use this Patient ID in xdstest2 to test the installation.

Remember these tools are an exact copy of what runs on the Public Registry server. Make sure you allocate for the correct registry.

Configuring the Registry/Repository

Some aspects of the Registry/Repository are configurable from a simple configuration table. Many others are horribly coded right into the Java. To edit the configuration table:

```
==> cd /usr/local/tomcat1/webapps/axis2/WEB-INF/services
==> mkdir work
==> cd work
==> jar -xf ../xds.aar
==> cat xds.properties
validate_patient_id=true
repository_base_dir=/usr/local/tomcat1/webapps/ROOT/Repository/
repository_base_uri=/Repository/
repository_port=9080
repository_machine_name=localhost
repository_unique_id=1.19.6.24.109.42.1.1
```

The two properties that are easy to alter are validate_patient_id which can be set to false and repository_unique_id which can be set to any value. The other properties are more tangled with the implementation. After updating the values, repackage by:

```
==> jar -cf ../xds.aar *
```

Restart tomcat1. It may auto-reload xds.aar but it may not. If you are watching catalina.out you will know how it behaves on your system.

Other areas of the Registry/Repository configuration use the code.xml table that is frequently distributed prior to Connectathons. This table resides at: /usr/local/tomcat1/webapps/xdsref/codes/codes.xml.

Note that if you want to change the Patient ID assigning authority, it is configured in two places:

1. In codes.xml at the end inside the element <AssigningAuthority/>
2. In the file tomcat1/webapps/xdstools/jsp/allocatePatientId.jsp in the line

```
<c:set target="${record}" property="patientIdAutoGenerated"
value="&1.3.6.1.4.1.21367.2005.3.7&ISO"/>
```

Note that in the first location it is XML encoded (& character) and in the second it is not.